# FENWICK & WEST LLP

SILICON VALLEY CENTER 801 CALIFORNIA STREET MOUNTAIN VIEW, CA 94041
TEL 650.988.8500 FAX 650.938.5200 WWW.FENWICK.COM

## FACSIMILE TRANSMISSION

## CONFIDENTIAL

**DATE:** January 26, 2010 **CLIENT-MATTER NUMBER:** 20423-10193

**To:**

| NAME: | FAX NO.: | PHONE NO.: |
|---|---|---|
| Examiner Ben Wang<br>USPTO | 571.270.2240 | 571.270-1240 |

**FROM:** Brian M. Hoffman **PHONE:** (415) 875-2484

**RE:** Proposed Examiner's Amendment for Serial No. 10/687,941

| NUMBER OF PAGES INCLUDING COVER PAGE: 12 | |
|---|---|

**MESSAGE:**

Please see attached.



24043/08537/SF/5285637.1

**IN THE UNITED STATES PATENT AND TRADEMARK OFFICE**

| | |
|---|---|
| APPLICANTS: | Abdul Kayam et al. |
| APPLICATION NO.: | 10/687,941 |
| FILING DATE: | October 17, 2003 |
| TITLE: | System and Associated Methods for Software Assembly |
| EXAMINER: | Ben C. Wang |
| GROUP ART UNIT: | 2192 |
| ATTY. DKT. NO.: | 24043-08537 |

---

MAIL STOP AMENDMENT
COMMISSIONER FOR PATENTS
P.O. BOX 1450
ALEXANDRIA, VA 22313-1450

## PROPOSED EXAMINER'S AMENDMENT

Attached is a proposed examiner's amendment.

## AMENDMENTS TO THE CLAIMS

1. (Currently Amended) A method of creating an application for executing on at least one machine having a memory, the method comprising:

creating a definition of at least one node and a specification, which are both held in at least one machine readable data file and written in a markup language;

the specification being arranged to be processed by a run time environment and the specification defining:

i: how the at least one node interacts with other nodes during the processing of the specification;

ii: resources useable by the at least one node during the processing of the specification;

iii: at least one set of predetermined rules used by the at least one node during the processing of the specification; and

iv: a set of messages which are arranged to be passed between nodes during the processing of the specification;

causing the run time environment to process the specification held in the machine readable data file such that the at least one node, as defined therein, is implemented within the memory of the machine thereby becoming a memory resident node;

the at least one memory resident node being arranged to:

i. receive messages as defined within the specification;

ii. process, according to rules defined in the specification for that node, data provided to the at least one memory resident node by messages such that rules are triggered if predetermined data is present within a message; and

iii. output further messages dependent upon triggering of rules within the node; and

creating the application by ~~the~~ processing of the specification, in the run time environment, such that ~~interconnecting the~~ at least one memory resident node is interconnected according to at least one of the specification and/~~or~~ data input, ~~such that~~ wherein data input to the application ~~created by processing of~~

~~the specification~~ is processed by the at least one memory resident node and, if further processing is required, forwarded to other nodes via links for that processing[[;]] ~~wherein~~ and links between nodes are dynamically configured responsive to amendments to the specification during processing thereof by the run time environment.

2. (Previously Presented) The method according to claim 1 in which a plurality of nodes are created.

3. (Previously Presented) The method according to claim 1 which further comprises providing a library of nodes containing at least one node and selecting at least one of the nodes from the library of nodes.

4. (Previously Presented) The method according to claim 1 which further comprises arranging the or each node to comprise a plurality of layers, each layer being arranged to perform a predetermined function.

5. (Previously Presented) The method according to claim 4 which further comprises arranging the layers of the nodes to be interchangeable and wherein altering at least one of the layers can change the overall functionality of a node.

6. (Previously Presented) The method according to claim 4 which further comprises providing a library of layers containing at least one layer and selecting at least one layer from the library of layers.

7. (Previously Presented) The method according to claim 4 which comprises arranging at least one of the layers of a node to act as a transport layer arranged to receive and send data to and from the node.

8. (Previously Presented) The method according to claim 4 which comprises arranging at least one of the layers of a node to act as a message transceiver arranged to send and receive messages to other nodes to which that node is connected.

9. (Previously Presented) The method according to claim 8 in which the at least one node has an identity and in which the application is arranged to be run and, at runtime, as nodes are connected together, the method further comprising arranging the message transceiver layer of a node to discover the identity of nodes to which it is connected at runtime.

10. (Previously Presented) The method according to claim 4 which comprises arranging at least one of the layers of a node to act as a rule processing engine arranged to apply the predetermined rules to data that the node receives.

11. (Previously Presented) The method according to claim 10 in which the rule processing engine layer of a node uses forward chaining rule logic.

12. (Previously Presented) The method according to claim 10 which comprises providing a rule set of at least one rule in a file that is used by the rule processing engine.

13. (Previously Presented) The method according to claim 12, which comprises specifying the file in which the rules are located by a link.

14. (Previously Presented) The method according to claim 10 which comprises defining each rule set that is to be used by the application in the specification.

15. Canceled.

16. Canceled.

17. (Currently Amended) The method according to claim 1 which comprises writing the messages in a flat text format, which may be any of the following: American Standard Code for Information Interchange (ASCII), Extensible Markup Language (XML), EDI (Electronic Data Interchange).

18. Canceled.

19. Canceled.

20. (Previously Presented) The method according to claim 1 which comprises providing a pattern, arranging the at least one node within the pattern and defining how nodes therein interact with one another.

21. (Previously Presented) The method according to claim 20, which comprises providing a library of patterns containing at least one pattern that can be used in creating an application.

22. (Previously Presented) The method according to claim 1 in which the specification is arranged to determine at least one of the following: which nodes are to be used; which nodes interact with one another; which patterns are to be used; which assets are to be used.

23. (Previously Presented) The method according to claim 1 which comprises providing files arranged to define the application specified therein, arranging the specification to be capable of deploying files and using the specification to deploy the files.

24. (Previously Presented) The method according to claim 23 which comprises arranging the files specifying the application to be XML files.

25. (Previously Presented) The method according to claim 1 in which the data processed by the application is specified in an XML file.

26. (Previously Presented) The method according to claim 1 in which data processed by the application is specified in an image file.

27. (Previously Presented) The method according to claim 1 in which data processed by the application is specified in a flat text file such as an ASCII) file, a raw text file, and EDI file.

28. (Previously Presented) The method according to claim 1 which comprises providing a graphical tool arranged to enable a user to specify components of the application.

29. (Previously Presented) The method according to claim 28 which comprises providing a library of at least one of the following: nodes; node layers; specification; patterns; messages; rule sets; style sheets; schemas and in which the graphical tool allows a user to select components from one of said libraries.

30. (Previously Presented) The method according to claim 29 which allows a user to define further libraries.

31. (Previously Presented) The method according to claim 28 which comprises providing at least one pattern arranged to define how nodes interact arranging the at least one pattern such that it is capable of interacting with at least one other pattern and arranging the graphical tool to allow a user to specify how the patterns and nodes interact with one another.

32. (Previously Presented) The method according to claim 28 which comprises using the graphical tool to perform at least one of the following: create the specification; edit the specification.

33. (Previously Presented) The method according to claim 28 which comprises using the graphical tool to manipulate any components of the specification.

34. (Previously Presented) The method according to claim 1 which comprises creating and deploying files and processing the files in the run time environment.

35. (Previously Presented) The method according to claim 1 in which at least one node is provided to provide an output from the application.

36. (Currently Amended) A computer system having a memory and being arranged to create an application, said system comprising:

a node creator arranged to create at least one machine readable data file containing a definition of at least one node and a specification, the definition and specification each written in a markup language;

the specification being arranged to be processed by a run time environment and the specification defining:

i: how the at least one node interacts with other nodes during the processing of the specification;

ii: resources useable by the at least one node during the processing of the specification;

iii: at least one set of predetermined rules used by the at least one node during the processing of the specification; and

iv: a set of messages which are arranged to be passed between that node and any other node during the processing of the specification;

a linker arranged to connect, dynamically according to at least one of the specification and/~~or~~ any data input to the application, at least two nodes such that data is arranged to pass between the nodes and the linker is arranged to interact with the node creator to modify the definition provided by the specification;

a deployer arranged to deploy the application from the definition created by the node creator and the linker according to the specification wherein the run time environment is arranged to implement the nodes in the memory of the computer system, the at least one node thereby becoming a memory resident node and the at least one memory resident node being arranged to process data according to the rules wherein at least one of the rules is triggered if predetermined data is present and an output is generated from that memory resident node.

37. (Previously Presented) The computer system according to claim 36 which comprises at least one processor arranged to process data, including the definition, and on which the definition created by the node creator and modified by the linker is processed.

38. (Previously Presented) The computer system according to claim 37 which comprises at least one processing apparatus comprising the at least one processor and in which the linker is arranged to connect nodes running on the processor within the processing apparatus.

39. (Previously Presented) The computer system according to claim 38 which comprises a plurality of processors, each remote from the other and having a connector therebetween capable of transmitting data between the processors.

40. (Previously Presented) The computer system according to claim 39 in which each of the processors is provided on a separate processing apparatus.

41. (Previously Presented) The computer system according to claim 39 in which the linker is arranged to connect nodes provided on processors remote from one another.

42. (Currently Amended) The computer system according to claim 36 in which the deployer deploys the definition that causes the nodes to communicate with one another using one of Hypertext Transfer Protocol (HTTP) and direct memory protocols.

43. (Previously Presented) The computer system according to claim 36 in which the node creator is arranged to utilise at least one of the following: predetermined definitions and pre-written definitions.

44. (Previously Presented) The computer system according to claim 43 in which the pre-written definition is provided in at least one library.

45. (Previously Presented) The computer system according to claim 36 which further comprises a pattern creator arranged to create at least one pattern of nodes.

46. (Previously Presented) The computer system according to claim 36 which further comprises a pattern cloner arranged to clone a pattern of nodes.

47. (Previously Presented) The computer system according to claim 36 which further comprises a rule creator arranged to allow predetermined rules to be created and edited.

48. (Previously Presented) The computer system according to claim 36 which further comprises at least one of the following: a node storage; a pattern storage; a rule storage.

49. (Currently Amended) A machine readable storage medium containing instructions which when read onto a computer cause that computer to perform the method of claim 1.

50. (Currently Amended) A machine readable storage medium containing instructions which when read onto a computer cause that computer to function as the computer system according to claim 36.

51. – 61. Canceled.

62. (Previously Presented) The method according to claim 1 in which the node, specification, and messages are at least in part written in XML.

63. (Currently Amended) A computer system having a memory and being arranged to run an application, said system comprising:

a run time environment arranged to process a definition of at least one node and a specification which are both written in a markup language and held within a machine readable data file, the specification defining:

i: how the at least one node interacts with other nodes during the processing of the specification;

ii: resources useable by the at least one node during the processing of the specification;

iii: at least one set of predetermined rules used by the at least one node during the processing of the specification; and

iv: a set of messages which are arranged to be passed between nodes during the processing of the specification;

wherein the run time environment is arranged to implement the at least one node in the memory of the computer system, the node thereby becoming a memory resident node and the at least one memory resident node being arranged to process data according to the rules wherein at least one of the rules is triggered if predetermined data is present and an output is generated from that memory resident node;

the run time environment also comprising a linker which is arranged to connect, ~~dynamically~~ according to at least one of the specification ~~and~~/or any data input to the application, the at least one node to any other nodes such that data input to the application is processed by the at least one node and, if further processing is required, forwarded to the other nodes for that further processing, such that links between nodes are dynamic responsive to amendments to the specification during processing thereof in the run time environment.

64. (Previously presented) A machine readable storage medium containing instructions which when read onto a computer cause that computer to function as the computer system according to claim 63.

65. (Previously presented) The method of claim 1, wherein the or each node is arranged to manipulate data contained in a message.

66. (Previously presented) The method of claim 65, wherein the or each node is arranged to manipulate XML data contained in the message.

67. (Previously presented) The method of claim 1, wherein the or each node is arranged, during processing of the machine readable data file, to output data to any of the nodes to which it is arranged to be connected.

68. (Previously Presented) The method of claim 1, wherein the run time environment directly processes the specification held in the machine readable data file.

69. (Previously Presented) The method of claim 1, wherein the run time environment processes the specification held in the machine readable data file without the need for extra processing of the specification.

70. (Previously Presented) The computer system of claim 36, wherein the run time environment is arranged to implement the nodes in the memory of the computer system without the need for extra processing of the specification.

71. (Previously Presented) The computer system of claim 63, wherein the run time environment is arranged to implement the at least one node in the memory of the computer system without the need for extra processing of the specification.

72. (Previously Presented) The method according to claim 1, wherein the specification and the data contained within the messages are written in the same language.

73. (Previously Presented) The method according to claim 1, wherein the rules implemented by the specification contain business logic.